Polynomial Acceleration

for STARK VMs

Alan Szepieniec
alan@neptune.cash

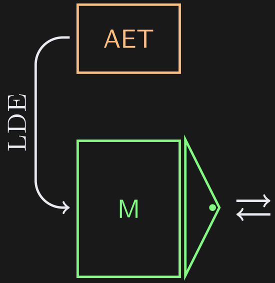
neptune.cash

triton-vm.org

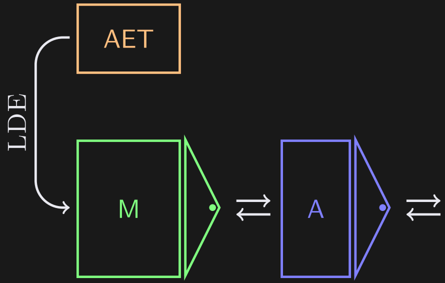
STARK Workflow with DEEP ALI

AET

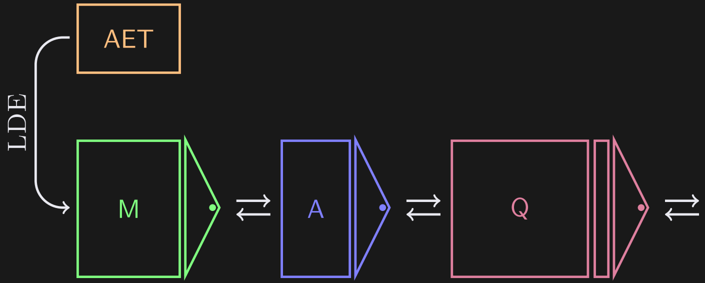
STARK Workflow with DEEP ALI



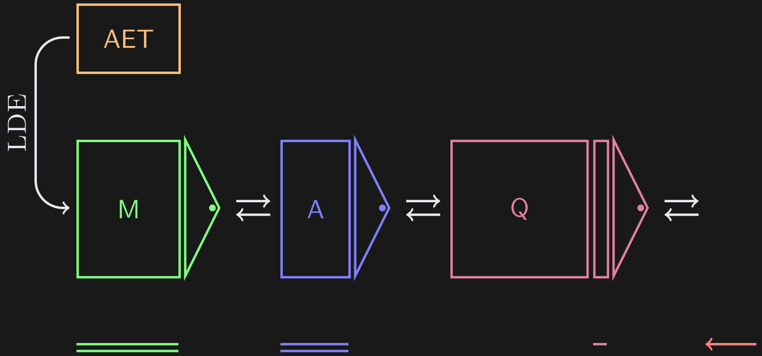
STARK Workflow with DEEP ALI



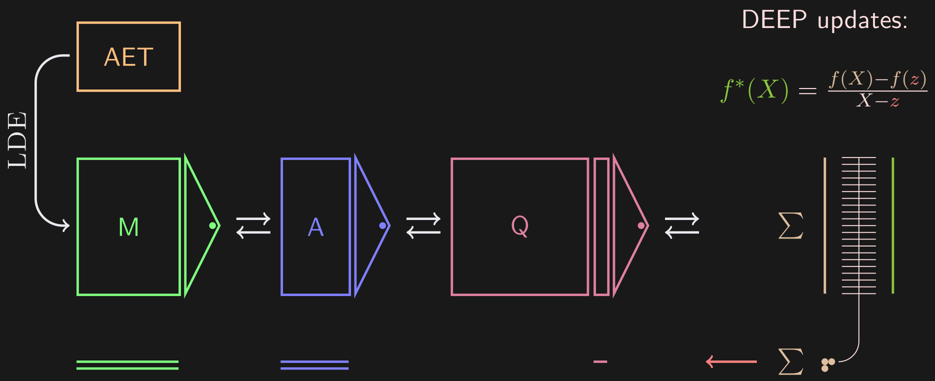
STARK Workflow with DEEP ALI



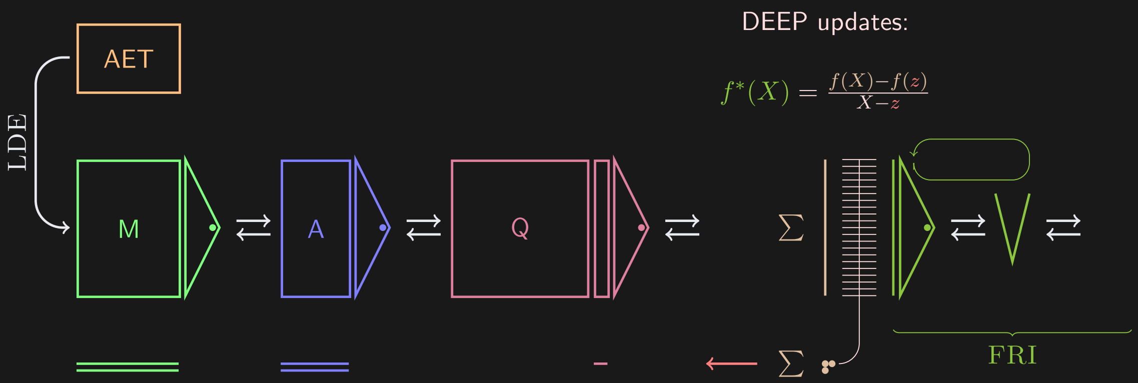
STARK Workflow with DEEP ALI



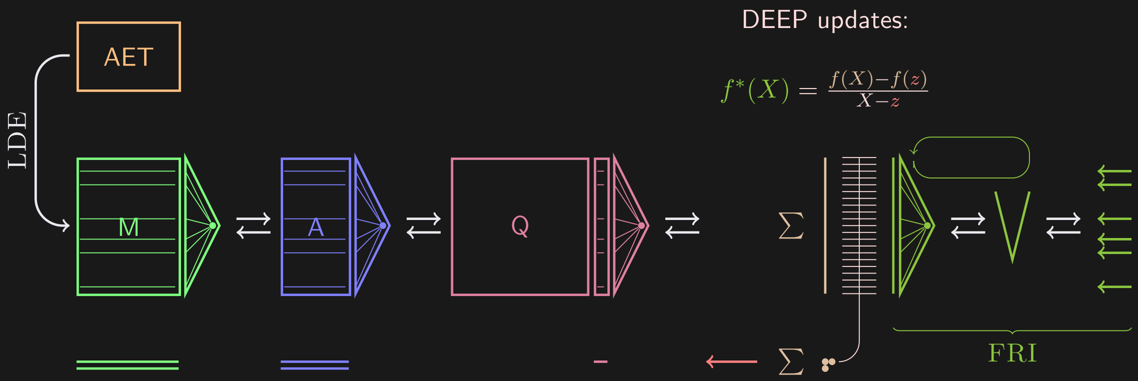
STARK Workflow with DEEP ALI



STARK Workflow with DEEP ALI



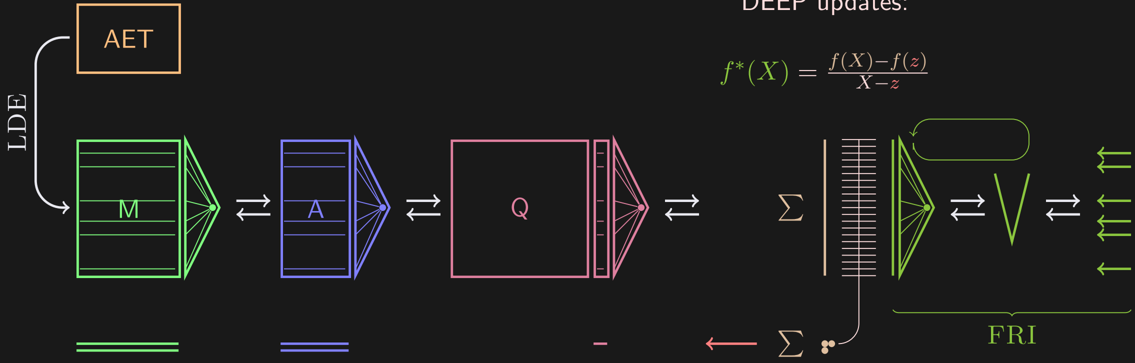
STARK Workflow with DEEP ALI



STARK Workflow with DEEP ALI

DEEP updates:

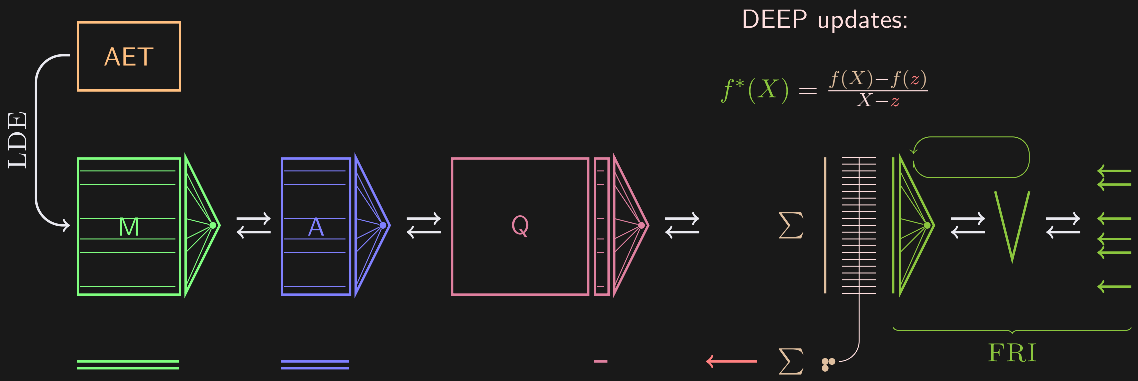
$$f^*(X) = \frac{f(X) - f(z)}{X - z}$$



TWO STAGES:

1. main
2. auxiliary

STARK Workflow with DEEP ALI



TWO STAGES:

1. main
2. auxiliary

Why not < 2?
How about > 2?

Two Stages

Two Stages

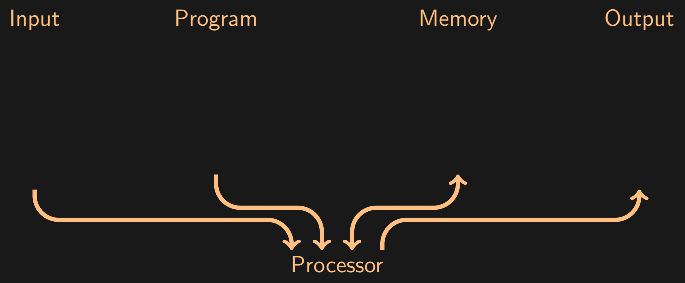
Processor

Two Stages

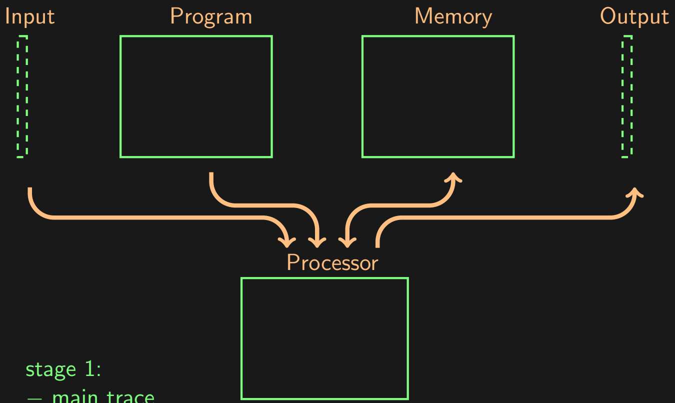
Input Program Memory Output

Processor

Two Stages



Two Stages

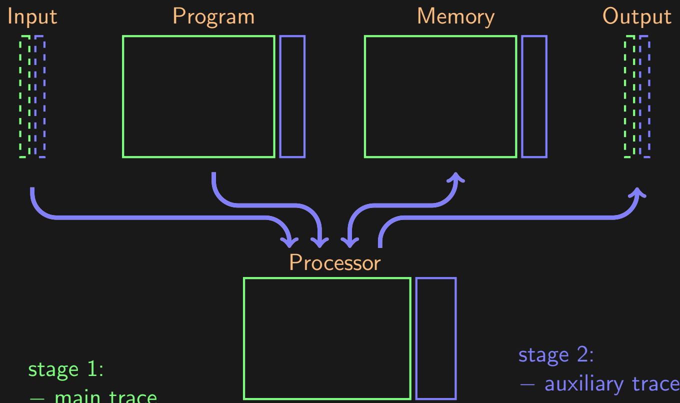


stage 1:

— main trace

— static

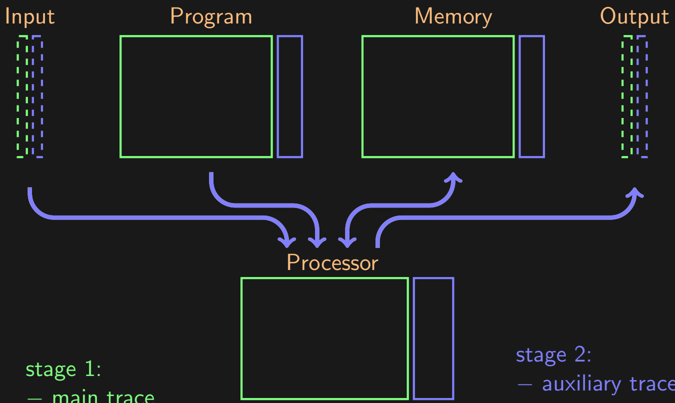
Two Stages



stage 1:
— main trace
— static

stage 2:
— auxiliary trace
— randomized
— cross-table arguments

Two Stages

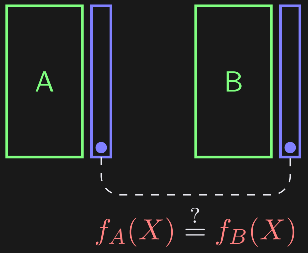


stage 1:
– main trace
– static

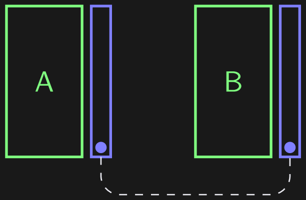
stage 2:
– auxiliary trace
– randomized
– cross-table arguments

Auxiliary columns { *link disparate logical units;*
depend on verifier randomness.

Cross-Table Arguments



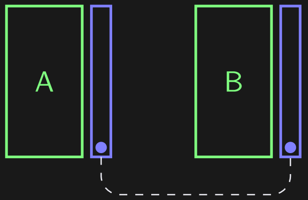
Cross-Table Arguments



$$f_A(X) \stackrel{?}{=} f_B(X)$$

order / multiplicity	argument	update rule
x / x	*lookup*	$f_A^{(i+1)}(X) = f_A^{(i)}(X) + \frac{\mathsf{mult}_{i+1}}{X - \mathsf{A}_{i+1}}$
x / ✓	*permutation*	$f_A^{(i+1)}(X) = f_A^{(i)}(X) \cdot (X - \mathsf{A}_{i+1})$
✓ / ✓	*evaluation*	$f_A^{(i+1)}(X) = f_A^{(i)}(X) \cdot X + \mathsf{A}_{i+1}$

Cross-Table Arguments

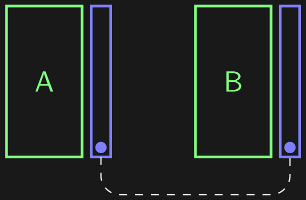


$$f_A(X) \stackrel{?}{=} f_B(X)$$

problem: AIR takes *scalars* not *polynomials*

order / multiplicity	argument	update rule
x / x	*lookup*	$f_A^{(i+1)}(X) = f_A^{(i)}(X) + \frac{\mathsf{mult}_{i+1}}{X - \mathsf{A}_{i+1}}$
x / ✓	*permutation*	$f_A^{(i+1)}(X) = f_A^{(i)}(X) \cdot (X - \mathsf{A}_{i+1})$
✓ / ✓	*evaluation*	$f_A^{(i+1)}(X) = f_A^{(i)}(X) \cdot X + \mathsf{A}_{i+1}$

Cross-Table Arguments



$$f_A(X) \stackrel{?}{=} f_B(X)$$

problem: AIR takes *scalars* not *polynomials*

solution: *collapse* polynomials to scalars

using *evaluation* in $\alpha \stackrel{\$}{\leftarrow} \mathbb{F}$

sound? Yes! Because Schwartz-Zippel.

order / multiplicity	argument	update rule
x / x	*lookup*	$f_A^{(i+1)}(X) = f_A^{(i)}(X) + \frac{\mathsf{mult}_{i+1}}{X - \mathsf{A}_{i+1}}$
x / ✓	*permutation*	$f_A^{(i+1)}(X) = f_A^{(i)}(X) \cdot (X - \mathsf{A}_{i+1})$
✓ / ✓	*evaluation*	$f_A^{(i+1)}(X) = f_A^{(i)}(X) \cdot X + \mathsf{A}_{i+1}$

Two Stages

1. Why > 1 stage?

Two Stages

1. Why > 1 stage?
   - $\rightarrow$ use elementary polynomial arithmetic to link tables
   - $\rightarrow$ collapse polynomials into scalars for AIR
   - $\rightarrow$ using *evaluation* in a *random point* $\alpha \stackrel{\$}{\leftarrow} \mathbb{F}$
   - $\rightarrow$ supplied by the *Verifier*

Two Stages

1. Why > 1 stage?
   - $\rightarrow$ use elementary polynomial arithmetic to link tables
   - $\rightarrow$ collapse polynomials into scalars for AIR
   - $\rightarrow$ using *evaluation* in a *random point* $\alpha \xleftarrow{\$} \mathbb{F}$
   - $\rightarrow$ supplied by the *Verifier*

2. How about > 2 stages?

Two Stages

1. Why > 1 stage?

- $\rightarrow$ use elementary polynomial arithmetic to link tables
- $\rightarrow$ collapse polynomials into scalars for AIR
- $\rightarrow$ using *evaluation* in a *random point* $\alpha \xleftarrow{\$} \mathbb{F}$
- $\rightarrow$ supplied by the *Verifier*

2. How about > 2 stages?

- $\rightarrow$ going from 1 stage $\rightarrow$ 2 stages unlocks power
- $\rightarrow$ going from 2 stages $\rightarrow$ 3 stages *unlocks which powers*?

Two Stages

1. Why > 1 stage?

   $\rightarrow$ use elementary polynomial arithmetic to link tables

   $\rightarrow$ collapse polynomials into scalars for AIR

   $\rightarrow$ using *evaluation* in a *random point* $\alpha \xleftarrow{\$} \mathbb{F}$

   $\rightarrow$ supplied by the *Verifier*

2. How about > 2 stages?

   $\rightarrow$ going from 1 stage $\rightarrow$ 2 stages unlocks power

   $\rightarrow$ going from 2 stages $\rightarrow$ 3 stages *unlocks which powers*?

   $\rightarrow$ *arbitrary* polynomial arithmetic

Virtual Machine State Evolution

traditional | polynomial acceleration | after substitution $X \mapsto \alpha$

instructions:

- basic arithmetic ✓
- add, multiply ✓
- polynomial commitment ✓
- polynomial division + remainder ✓
- polynomial evaluation queries ✓
- degree bound checks ✓

Application 1: Big Integer Arithmetic

$$a \times b$$

$a, b \in \mathbb{N}$ and $a, b \gg p$

1. construct $a(X), b(X), c(X)$ with coefficients $\in \{0, 1\}$
2. calculate $r(X) = (a(X) \cdot b(X) - c(X))\,\%(X - 2)$
3. assert $r(X) = 0$

Application 1: Big Integer Arithmetic

$$a \times b$$

$a, b \in \mathbb{N}$ and $a, b \gg p$

1. construct $a(X), b(X), c(X)$ with coefficients $\in \{0, 1\}$
2. calculate $r(X) = (a(X) \cdot b(X) - c(X))\,\%(X - 2)$
3. assert $r(X) = 0$

$a(X) \cdot b(X) \equiv c(X) \mod X - 2$ (assumes no coefficient overflow)

Application 1: Big Integer Arithmetic

$$a \times b$$

$a, b \in \mathbb{N}$ and $a, b \gg p$

cycles:

linear	1. construct $a(X), b(X), c(X)$ with coefficients $\in \{0, 1\}$
constant	2. calculate $r(X) = (a(X) \cdot b(X) - c(X)) \,\%(X - 2)$
constant	3. assert $r(X) = 0$

$a(X) \cdot b(X) \equiv c(X) \mod X - 2$ (assumes no coefficient overflow)

Application 1: Big Integer Arithmetic

$$a \times b$$

$a, b \in \mathbb{N}$ and $a, b \gg p$

 # cycles:

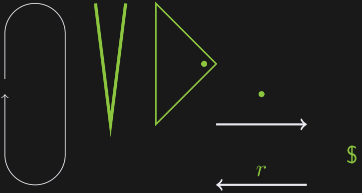
# cycles	
linear	1. construct $a(X), b(X), c(X)$ with coefficients $\in \{0, 1\}$
constant	2. calculate $r(X) = (a(X) \cdot b(X) - c(X)) \,\%(X - 2)$
constant	3. assert $r(X) = 0$

(independent of circuit depth)

$a(X) \cdot b(X) \equiv c(X) \mod X - 2$ (assumes no coefficient overflow)

Application 2: Last Codeword in FRI

many rounds:



last round:

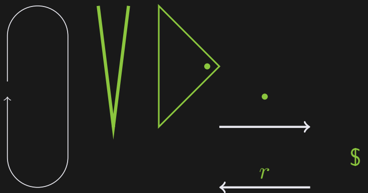
1. Horner evaluation
2. barycentric extrapolation
3. t colinearity checks

polynomial acceleration:

1. construct polynomial
2. evaluate t times
3. t colinearity checks

Application 2: Last Codeword in FRI

many rounds:



last round:

polynomial acceleration:

	# cycles:
1. Horner evaluation	$O(d)$
2. barycentric extrapolation	$O(\rho^{-1} \cdot d)$
3. t colinearity checks	$O(t)$
1. construct polynomial	$O(d)$
2. evaluate t times	$O(t)$
3. t colinearity checks	$O(t)$

Application 3: SIMD

$$a(X) + b(X) \quad \Leftrightarrow \quad \mathbf{a} + \mathbf{b}$$

Application 3: SIMD

$$a(X) + b(X) \quad \Leftrightarrow \quad \mathbf{a} + \mathbf{b}$$

$$a(X) \times b(X) \,\%\, X^N - 1 \quad \Leftrightarrow \quad \mathbf{a} \circ \mathbf{b}$$

Application 3: SIMD

$$a(X) + b(X) \quad \Leftrightarrow \quad \mathbf{a} + \mathbf{b}$$

$$a(X) \times b(X) \,\%\, X^N - 1 \quad \Leftrightarrow \quad \mathbf{a} \circ \mathbf{b}$$

$$a(\omega^i) \quad \Leftrightarrow \quad a_i$$

Application 3: SIMD

$$a(X) + b(X) \quad \Leftrightarrow \quad \mathbf{a} + \mathbf{b}$$

$$a(X) \times b(X) \,\%\, X^N - 1 \quad \Leftrightarrow \quad \mathbf{a} \circ \mathbf{b}$$

$$a(\omega^i) \quad \Leftrightarrow \quad a_i$$

$$s \leftarrow \sum_{x \in \langle \omega \rangle} a(x) \quad \Leftrightarrow \quad s \leftarrow \sum_i a_i$$

$\rightarrow$ still need to assert

$$\exists\, c(X) : a(X) - \frac{s}{|\langle \omega \rangle|} = c(X) - c(\omega X)$$

Application 3: SIMD

linear construction $\rightarrow$ $a(X) + b(X) \Leftrightarrow \mathbf{a} + \mathbf{b}$

$a(X) \times b(X) \,\%\, X^N - 1 \Leftrightarrow \mathbf{a} \circ \mathbf{b}$

$a(\omega^i) \Leftrightarrow a_i$

$s \leftarrow \sum_{x \in \langle \omega \rangle} a(x) \Leftrightarrow s \leftarrow \sum_i a_i$

$\rightarrow$ still need to assert

batchable $\rightarrow$ $\exists\, c(X) : a(X) - \frac{s}{|\langle \omega \rangle|} = c(X) - c(\omega X)$

Construction

pr1	pr1d	···	prn	prnd	r1	r1d	r2	r2d	**acc**	ninv	dinv	qinv	as1	···	as5

Construction

pr1	pr1d	⋯	prn	prnd	r1	r1d	r2	r2d	acc	ninv	dinv	qinv	as1	⋯	as5
$n(X)$	X^{-b_n}		$d(X)$	X^{-b_d}	$q(X)$	X^{-b_q}	$r(X)$	X^{-b_r}	$a(X,Y)$	X^{b_n}	X^{b_d}	X^{b_q}			

Construction

pr1	pr1d	···	prn	prnd	r1	r1d	r2	r2d	acc	ninv	dinv	qinv	as1	···	as5
$n(X)$	X^{-b_n}		$d(X)$	X^{-b_d}	$q(X)$	X^{-b_q}	$r(X)$	X^{-b_r}	$a(X,Y)$	X^{b_n}	X^{b_d}	X^{b_q}	$a_{-1}(X,\beta)$		

Construction

pr1	pr1d	$\cdots$	prn	prnd	r1	r1d	r2	r2d	acc	ninv	dinv	qinv	as1	$\cdots$	as5
$n(X)$	X^{-b_n}		$d(X)$	X^{-b_d}	$q(X)$	X^{-b_q}	$r(X)$	X^{-b_r}	$a(X,Y)$	X^{b_n}	X^{b_d}	X^{b_q}	$a_{-1}(X,\beta)$		
$n(\alpha)$	α^{-b_n}		$d(\alpha)$	α^{-b_d}	$q(\alpha)$	α^{-b_q}	$r(\alpha)$	α^{-b_r}	$a(\alpha,\beta)$	α^{b_n}	α^{b_d}	α^{b_q}			

Construction

pr1	pr1d	$\cdots$	prn	prnd	r1	r1d	r2	r2d	acc	ninv	dinv	qinv	as1	$\cdots$	as5
$n(X)$	X^{-b_n}		$d(X)$	X^{-b_d}	$q(X)$	X^{-b_q}	$r(X)$	X^{-b_r}	$a(X,Y)$	X^{b_n}	X^{b_d}	X^{b_q}	$a_{-1}(X,\beta)$		
$n(\alpha)$	α^{-b_n}		$d(\alpha)$	α^{-b_d}	$q(\alpha)$	α^{-b_q}	$r(\alpha)$	α^{-b_r}	$a(\alpha,\beta)$	α^{b_n}	α^{b_d}	α^{b_q}			

div-mod: $q(X) \leftarrow n(X) \,//\, d(X) \quad r(X) \leftarrow n(X) \,\%\, d(X)$

Construction

pr1	pr1d	$\cdots$	prn	prnd	r1	r1d	r2	r2d	acc	ninv	dinv	qinv	as1	$\cdots$	as5
$n(X)$	X^{-b_n}		$d(X)$	X^{-b_d}	$q(X)$	X^{-b_q}	$r(X)$	X^{-b_r}	$a(X,Y)$	X^{b_n}	X^{b_d}	X^{b_q}	$a_{-1}(X,\beta)$		
$n(\alpha)$	α^{-b_n}		$d(\alpha)$	α^{-b_d}	$q(\alpha)$	α^{-b_q}	$r(\alpha)$	α^{-b_r}	$a(\alpha,\beta)$	α^{b_n}	α^{b_d}	α^{b_q}			

div-mod: $q(X) \leftarrow n(X) \,//\, d(X) \quad r(X) \leftarrow n(X) \,\%\, d(X)$

$n(X) = d(X) \cdot q(X) + r(X)$ $\downarrow$ AIR ✓

$\deg(r) < \deg(d)$ $\downarrow$

$\deg(d) + \deg(q) = \deg(n)$ $\downarrow$

Construction

pr1	pr1d	⋯	prn	prnd	r1	r1d	r2	r2d	acc	ninv	dinv	qinv	as1	⋯	as5
$n(X)$	X^{-b_n}		$d(X)$	X^{-b_d}	$q(X)$	X^{-b_q}	$r(X)$	X^{-b_r}	$a(X,Y)$	X^{b_n}	X^{b_d}	X^{b_q}	$a_{-1}(X,\beta)$		
$n(\alpha)$	α^{-b_n}		$d(\alpha)$	α^{-b_d}	$q(\alpha)$	α^{-b_q}	$r(\alpha)$	α^{-b_r}	$a(\alpha,\beta)$	α^{b_n}	α^{b_d}	α^{b_q}			

div-mod: $q(X) \leftarrow n(X) \,//\, d(X) \quad r(X) \leftarrow n(X) \,\%\, d(X)$

$$n(X) = d(X) \cdot q(X) + r(X)$$

$$\downarrow$$

AIR ✓

$$\deg(r) < \deg(d)$$

$$\downarrow$$

$$a \mapsto Y \cdot a + X^N \cdot X^{-b_r} \cdot r(X)$$

$$a \mapsto Y \cdot a + X^{b_d - b_r - 1}$$

$$\deg(d) + \deg(q) = \deg(n)$$

$$\downarrow$$

$$a \mapsto Y \cdot a + X^N \cdot X^{-b_q} \cdot q(X)$$

$$a \mapsto Y \cdot a + X^{b_n - b_q - b_d}$$

$$a \mapsto Y \cdot a + X^{b_q + b_d - b_n}$$

Construction

pr1	pr1d	$\cdots$	prn	prnd	r1	r1d	r2	r2d	**acc**	ninv	dinv	qinv	as1	$\cdots$	as5
$n(X)$	X^{-b_n}		$d(X)$	X^{-b_d}	$q(X)$	X^{-b_q}	$r(X)$	X^{-b_r}	$a(X,Y)$	X^{b_n}	X^{b_d}	X^{b_q}	$a_{-1}(X,\beta)$		
$n(\alpha)$	α^{-b_n}		$d(\alpha)$	α^{-b_d}	$q(\alpha)$	α^{-b_q}	$r(\alpha)$	α^{-b_r}	$a(\alpha,\beta)$	α^{b_n}	α^{b_d}	α^{b_q}			

div-mod: $q(X) \leftarrow n(X) \,//\, d(X) \quad r(X) \leftarrow n(X) \,\%\, d(X)$

$$n(X) = d(X) \cdot q(X) + r(X)$$

↓

AIR ✓

$$\deg(r) < \deg(d)$$

↓

$$a \mapsto Y \cdot a + X^N \cdot X^{-b_r} \cdot r(X)$$

$$a \mapsto Y \cdot a + X^{b_d - b_r - 1}$$

↓

AIR ✓

$$\deg(d) + \deg(q) = \deg(n)$$

↓

$$a \mapsto Y \cdot a + X^N \cdot X^{-b_q} \cdot q(X)$$

$$a \mapsto Y \cdot a + X^{b_n - b_q - b_d}$$

$$a \mapsto Y \cdot a + X^{b_q + b_d - b_n}$$

→ AIR ✓

Construction

pr1	pr1d	⋯	prn	prnd	r1	r1d	r2	r2d	**acc**	ninv	dinv	qinv	as1	⋯	as5
$n(X)$	X^{-b_n}		$d(X)$	X^{-b_d}	$q(X)$	X^{-b_q}	$r(X)$	X^{-b_r}	$a(X,Y)$	X^{b_n}	X^{b_d}	X^{b_q}	$a_{-1}(X,\beta)$		
$n(\alpha)$	α^{-b_n}		$d(\alpha)$	α^{-b_d}	$q(\alpha)$	α^{-b_q}	$r(\alpha)$	α^{-b_r}	$a(\alpha,\beta)$	α^{b_n}	α^{b_d}	α^{b_q}			

div-mod: $q(X) \leftarrow n(X) \,//\, d(X) \quad r(X) \leftarrow n(X) \,\%\, d(X)$

$$n(X) = d(X) \cdot q(X) + r(X)$$

$\downarrow$

AIR ✓

$$\deg(r) < \deg(d)$$

$\downarrow$

$$a \mapsto Y \cdot a + X^N \cdot X^{-b_r} \cdot r(X)$$

$$a \mapsto Y \cdot a + X^{b_d - b_r - 1}$$

$\downarrow$

AIR ✓

$$\deg(d) + \deg(q) = \deg(n)$$

$\downarrow$

$$a \mapsto Y \cdot a + X^N \cdot X^{-b_q} \cdot q(X)$$

$$a \mapsto Y \cdot a + X^{b_n - b_q - b_d}$$

$$a \mapsto Y \cdot a + X^{b_q + b_d - b_n}$$

$\rightarrow$ AIR ✓

$$\deg_X(a_{-1}(X,Y)) \le N \overset{\text{SZL}}{\Leftrightarrow} \deg(a_{-1}(X,\beta)) \le N$$

Construction

pr1	pr1d	⋯	prn	prnd	r1	r1d	r2	r2d	**acc**	ninv	dinv	qinv	as1	⋯	as5
$n(X)$	X^{-b_n}		$d(X)$	X^{-b_d}	$q(X)$	X^{-b_q}	$r(X)$	X^{-b_r}	$a(X,Y)$	X^{b_n}	X^{b_d}	X^{b_q}	$a_{-1}(X,\beta)$		
$n(\alpha)$	α^{-b_n}		$d(\alpha)$	α^{-b_d}	$q(\alpha)$	α^{-b_q}	$r(\alpha)$	α^{-b_r}	$a(\alpha,\beta)$	α^{b_n}	α^{b_d}	α^{b_q}			

FRI: $a_{-1}(X,\beta) \Downarrow \deg(a_{-1}(X,\beta)) \le N$

div-mod: $q(X) \leftarrow n(X) \,//\, d(X)$ $\quad r(X) \leftarrow n(X) \,\%\, d(X)$

$n(X) = d(X) \cdot q(X) + r(X)$ $\downarrow$ AIR ✓

$\deg(r) < \deg(d)$

$\downarrow$

$a \mapsto Y \cdot a + X^N \cdot X^{-b_r} \cdot r(X)$

$a \mapsto Y \cdot a + X^{b_d - b_r - 1}$

$\downarrow$

AIR ✓

$\deg(d) + \deg(q) = \deg(n)$

$\downarrow$

$a \mapsto Y \cdot a + X^N \cdot X^{-b_q} \cdot q(X)$

$a \mapsto Y \cdot a + X^{b_n - b_q - b_d}$

$a \mapsto Y \cdot a + X^{b_q + b_d - b_n}$

$\rightarrow$ AIR ✓

$\deg(a_{-1}(X,\beta)) \le N$

$$\deg_X(a_{-1}(X,Y)) \le N \overset{\text{SZL}}{\Leftrightarrow} \deg(a_{-1}(X,\beta)) \le N$$

Construction

pr1	pr1d	⋯	prn	prnd	r1	r1d	r2	r2d	acc	ninv	dinv	qinv	as1	⋯	as5
$n(X)$	X^{-b_n}		$d(X)$	X^{-b_d}	$q(X)$	X^{-b_q}	$r(X)$	X^{-b_r}	$a(X,Y)$	X^{b_n}	X^{b_d}	X^{b_q}	$a_{-1}(X,\beta)$		
$n(\alpha)$	α^{-b_n}		$d(\alpha)$	α^{-b_d}	$q(\alpha)$	α^{-b_q}	$r(\alpha)$	α^{-b_r}	$a(\alpha,\beta)$	α^{b_n}	α^{b_d}	α^{b_q}			

FRI ↓

div-mod: $q(X) \leftarrow n(X) \,//\, d(X) \quad r(X) \leftarrow n(X) \,\%\, d(X)$

$n(X) = d(X) \cdot q(X) + r(X)$ ↓ AIR ✓

$\deg(r) < \deg(d)$ ↓

$a \mapsto Y \cdot a + X^N \cdot X^{-b_r} \cdot r(X)$

$a \mapsto Y \cdot a + X^{b_d - b_r - 1}$

↓ AIR ✓

$\deg(d) + \deg(q) = \deg(n)$ ↓

$a \mapsto Y \cdot a + X^N \cdot X^{-b_q} \cdot q(X)$

$a \mapsto Y \cdot a + X^{b_n - b_q - b_d}$

$a \mapsto Y \cdot a + X^{b_q + b_d - b_n}$

→ AIR ✓

$\deg(a_{-1}(X,\beta)) \leq N$

$a(\alpha,\beta)$

$\deg_X(a_{-1}(X,Y)) \leq N \overset{\text{SZL}}{\Leftrightarrow} \deg(a_{-1}(X,\beta)) \leq N$

Construction

pr1	pr1d	$\cdots$	prn	prnd	r1	r1d	r2	r2d	acc	ninv	dinv	qinv	as1	$\cdots$	as5
$n(X)$	X^{-b_n}		$d(X)$	X^{-b_d}	$q(X)$	X^{-b_q}	$r(X)$	X^{-b_r}	$a(X,Y)$	X^{b_n}	X^{b_d}	X^{b_q}	$a_{-1}(X,\beta)$		
$n(\alpha)$	α^{-b_n}		$d(\alpha)$	α^{-b_d}	$q(\alpha)$	α^{-b_q}	$r(\alpha)$	α^{-b_r}	$a(\alpha,\beta)$	α^{b_n}	α^{b_d}	α^{b_q}			

FRI

div-mod: $q(X) \leftarrow n(X) \,//\, d(X) \quad r(X) \leftarrow n(X) \,\%\, d(X)$

$n(X) = d(X) \cdot q(X) + r(X)$

$\downarrow$

AIR ✓

$\deg(r) < \deg(d)$

$\downarrow$

$a \mapsto Y \cdot a + X^N \cdot X^{-b_r} \cdot r(X)$

$a \mapsto Y \cdot a + X^{b_d - b_r - 1}$

$\downarrow$

AIR ✓

$\deg(d) + \deg(q) = \deg(n)$

$\downarrow$

$a \mapsto Y \cdot a + X^N \cdot X^{-b_q} \cdot q(X)$

$a \mapsto Y \cdot a + X^{b_n - b_q - b_d}$

$a \mapsto Y \cdot a + X^{b_q + b_d - b_n}$

$\deg(a_{-1}(X,\beta)) \leq N$

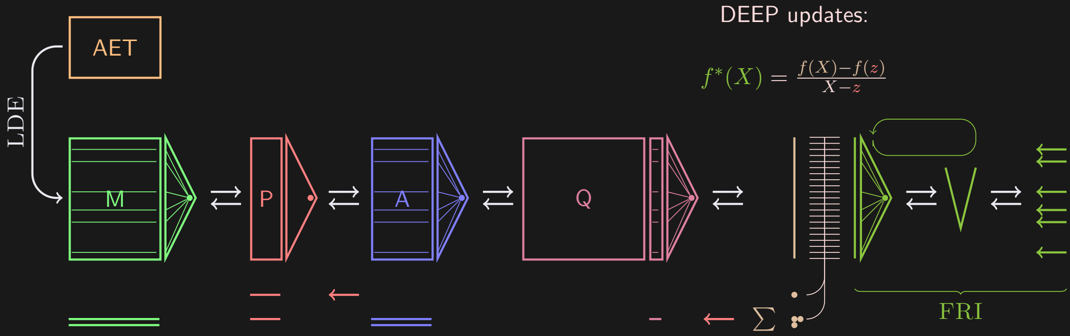
$\updownarrow$

- OODE in α
- DEEP update
- AIR ✓

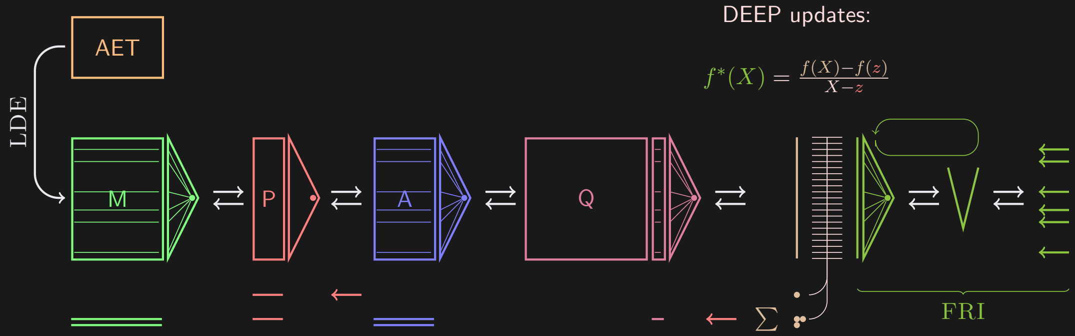
$a(\alpha,\beta)$

$$\deg_X(a_{-1}(X,Y)) \leq N \overset{\text{SZL}}{\Leftrightarrow} \deg(a_{-1}(X,\beta)) \leq N$$

STARK Workflow with DEEP ALI and Polynomial Acceleration



STARK Workflow with DEEP ALI and Polynomial Acceleration



THREE STAGES:

1. main
2. accumulator polynomial segments
3. auxiliary

Limitations

1. No Composition
   - AIR applies to *scalars*
   - no scaling or sign flips
   - dihedral permutations *or* SIMD multiply
   - *can* verify compositions but ...

Limitations

1. No Composition
   - AIR applies to *scalars*
   - no scaling or sign flips
   - dihedral permutations *or* SIMD multiply
   - *can* verify compositions but ...

2. No Divination
   - no non-determinism
   - VM runs on *prover-side* not *verifier-side*
   - decommitment takes *linear* # AET cells

Open Problem

Sublinear Construction

Open Problem

Sublinear Construction

– verify that a *divined* polynomial of degree N

corresponds uniquely* to a constant-size cryptographic digest

in $< O(N)$ cycles

using polynomial acceleration instructions (including SIMD)

*: up to cryptographic hardness assumptions

Open Problem

Sublinear Construction

- verify that a *divined* polynomial of degree N

  corresponds uniquely* to a constant-size cryptographic digest

  in $< O(N)$ cycles

  using polynomial acceleration instructions (including SIMD)

*: up to cryptographic hardness assumptions

Achieves:

- simulate arbitrary Polynomial IOPs *as prover*

  $\rightarrow$ shifts cost to *outside of AET*